OTUS

Онлайн образование

otus.ru

Проверить, идет ли запись

Меня хорошо видно && слышно?

Тема вебинара

Введение в docker. Обзор docker-compose

Золотов Дмитрий

Flutter & Kotlin Developer @ Yandex, DevOps, Fullstack Developer

Преподаватель

Золотов Дмитрий

Full-Stack разработчик (Python / Java / Kotlin / Flutter) - более 20 лет опыта.
В Kotlin с 2015 года (backend, JS, кроссплатформа)
DevOps и консультант по миграции высонагруженных систем в гибридные облака
Приглашенный преподаватель ИТМО
Ранее руководитель отдела автоматизации в финансовой организации
Разработчик в Яндекс (Flutter)

Правила вебинара

Активно
участвуем

Off-topic обсуждаем
в Telegram

Задаем вопрос
в чат или голосом

Вопросы вижу в чате,
могу ответить не сразу

Условные обозначения

Индивидуально

Время, необходимое
на активность

Пишем в чат

Говорим голосом

Документ

Ответьте себе или
задайте вопрос

Маршрут вебинара

- Знакомство
- Docker: концепции
- Dockerfile
- Docker: сети
- Docker-compose
- Рефлексия

Цели вебинара

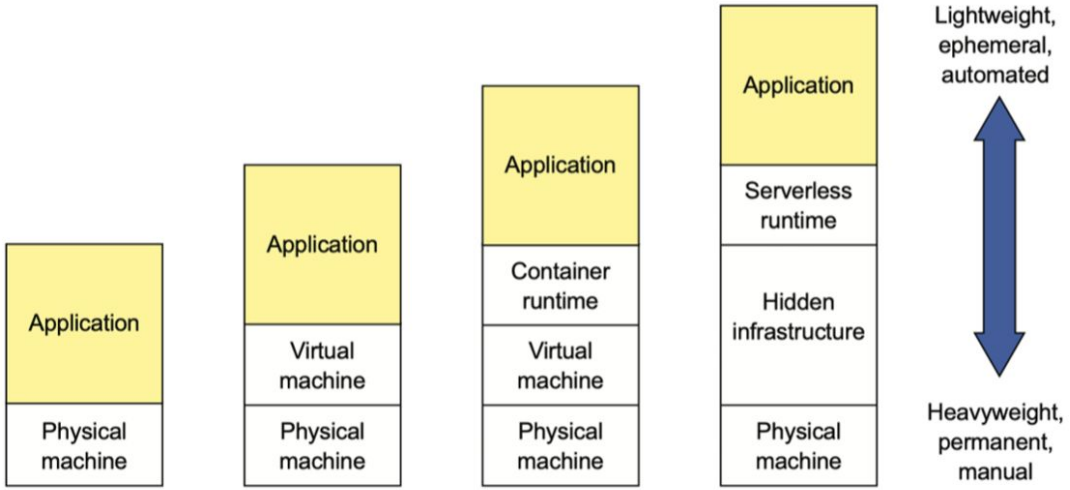
После занятия вы сможете

1. Обертывать сервисы в docker
2. Разворачивать consul

Docker: концепции

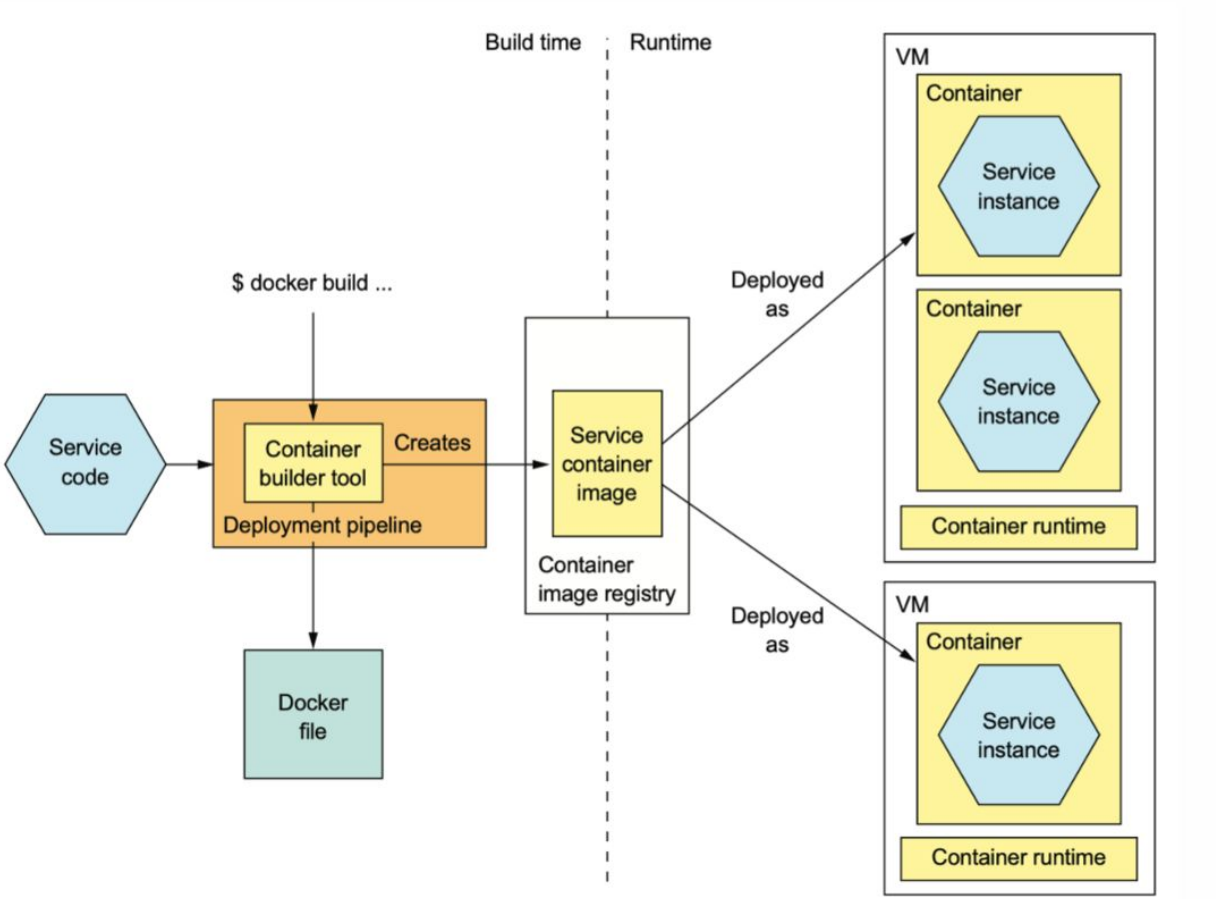
Как разместить несколько сервисов на одной машине?

- **Сервер приложений** jvm (tomcat/jboss), python uwsgi
- **Виртуальные машины** (Vagrant, vmware)
- **Контейнеры** (Docker, rkt)



Контейнеры

- Technology agnostic
- Изоляция и ограничение сервисов друг от друга (cgroups)
- Эффективная утилизация ресурсов



Overhead контейнеров

https://domino.research.ibm.com/library/cyberdig.nsf/papers/092905219 5DD819C85257D2300681E7B/$File/rc25482.pdf – исследование сравнения производительности Docker контейнеров.

- Оверхед в основном по сети (NAT)
- По диску и CPU docker container практически идентичен нативному исполнению

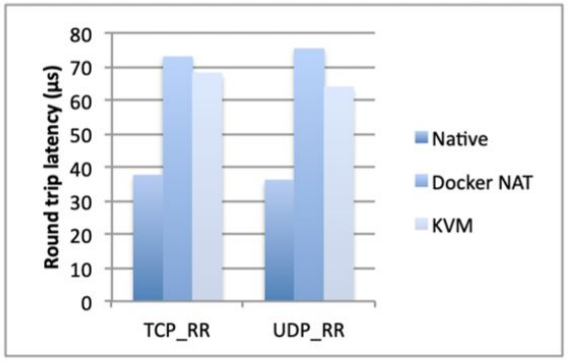


Fig. 3. Network round-trip latency (μs).

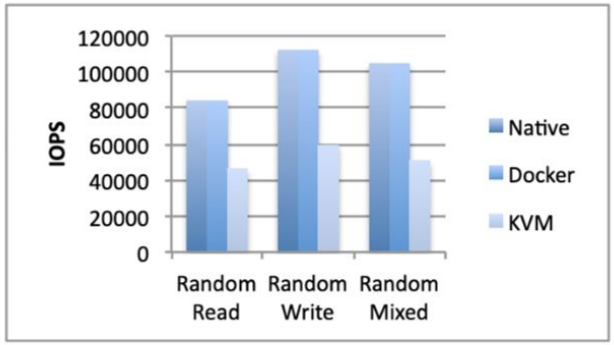


Fig. 6. Random I/O throughput (IOPS).

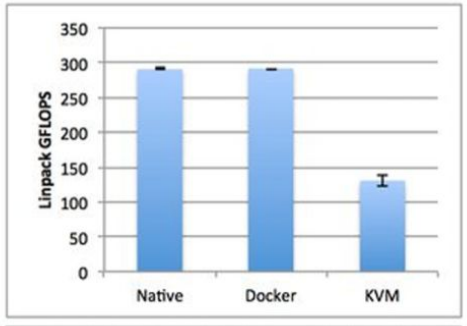


Figure 1. Linpack performance on two sockets (16 cores). Each data point is the arithmetic mean obtained from ten runs. Error bars indicate the standard deviation obtained over all runs.

Docker

- Существовало достаточно давно
- Не было широкого распространения
- В определенных случаях была заменена аппаратная виртуализация
- Не столько про контейнеры (как технология)

Шпаргалка с командами Docker / Хабр

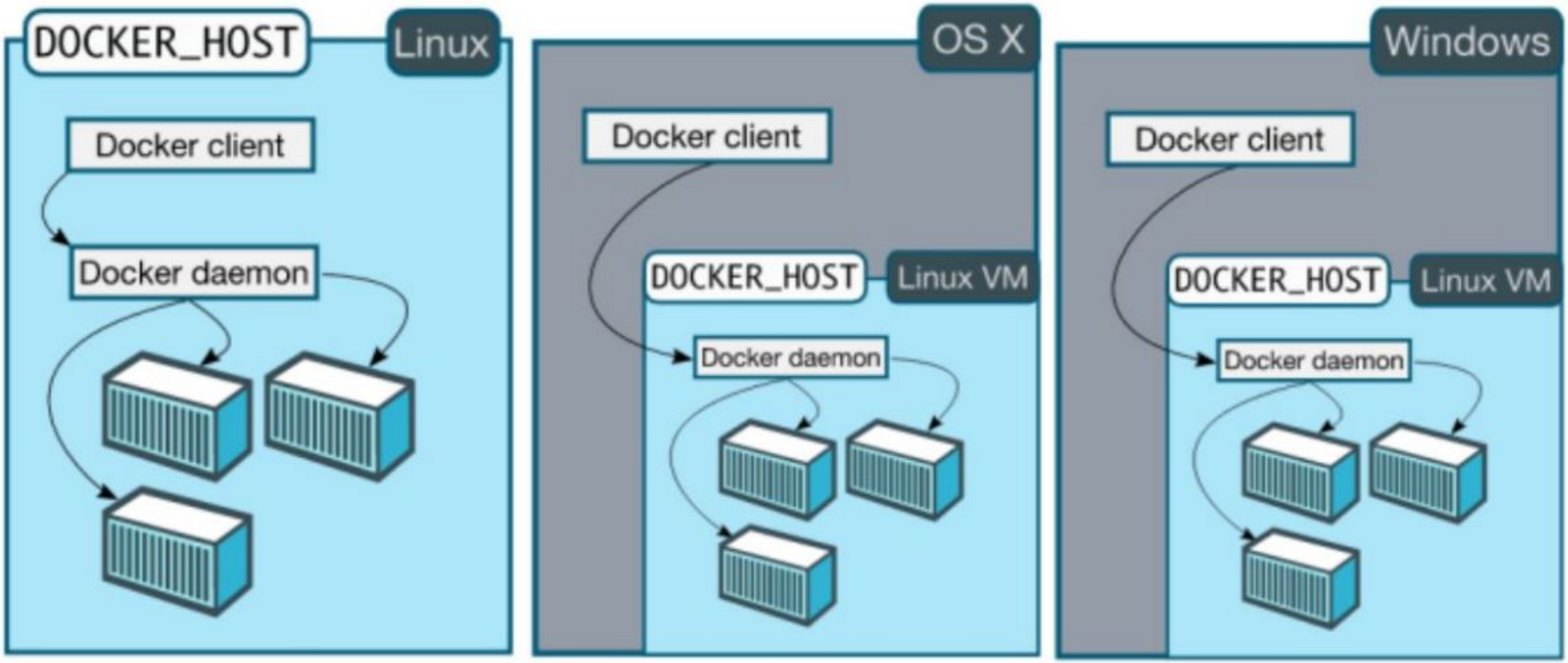
Docker

- Абстракция от host-системы
- Легковесные изолированные окружения
- Общие слои файловой системы
- Компоновка и предсказуемость
- Простое управление зависимостями
- Дистрибьюция и тиражируемость
- Стандартизация описания окружения, сборки
- 100% консистентная среда приложения
- Воспроизводимость

Docker

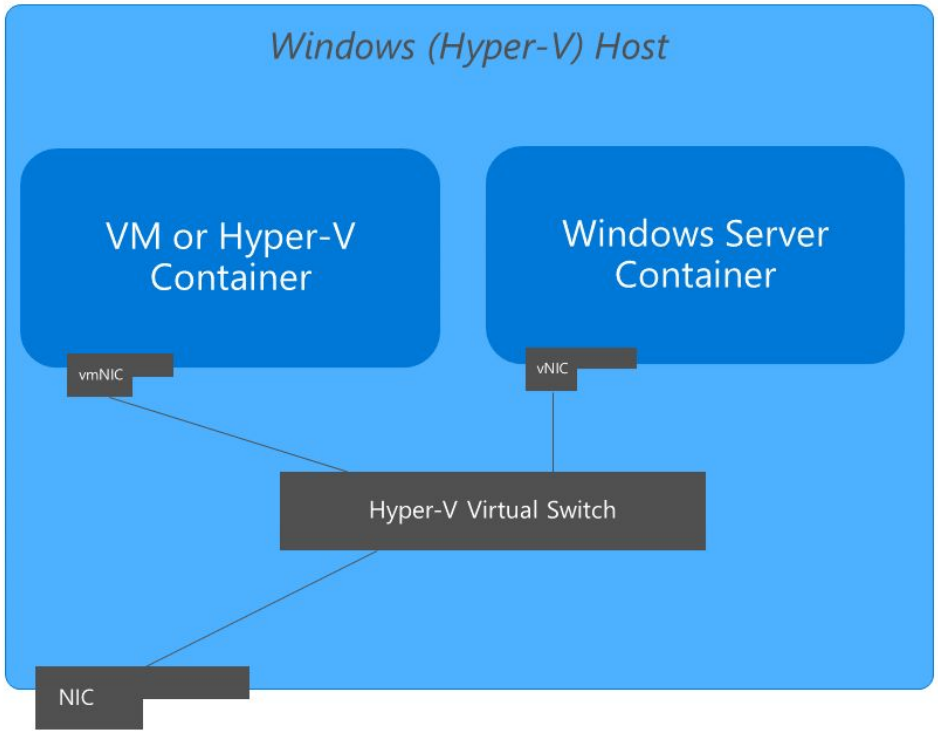
Docker - это не виртуальная машина!

Docker


DOCKER_HOST
Linux
Docker client
Docker daemon
OS X
Docker client
DOCKER_HOST
Linux VM
Docker daemon
Windows
Docker client
DOCKER_HOST
Linux VM
Docker daemon

Windows containers

About Windows containers | Microsoft Learn
Prep Windows operating system containers | Microsoft Learn
Windows Container Base Images | Microsoft Learn
Windows container networking | Microsoft Learn



Docker

Принцип работы:

- Namespaces
- Cgroups
- UnionFS
- RunC (Управление контейнерами с runC / Хабр)

Docker Namespaces

- Изоляция окружения
- Индивидуальный namespace для каждого контейнера
- Pid, net, mount, etc.
- Namespace прекращает свое существование после окончания работы PID1

https://docs.docker.com/engine/security/userns-remap/

Understanding and Securing Linux Namespaces

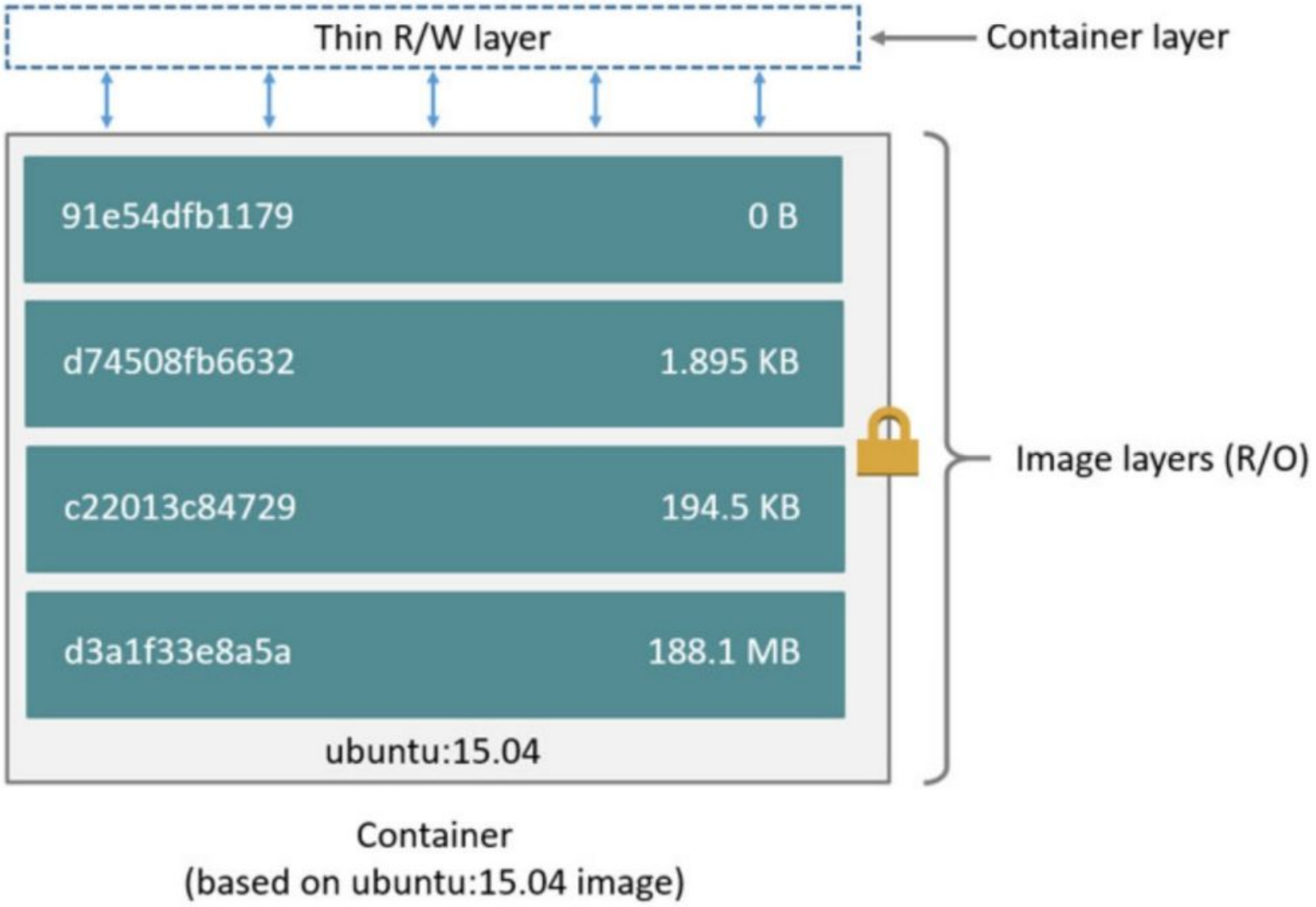
Docker cgroups

- Использование контейнерами общих ресурсов
- Ограничение ресурсов
- CPU, memory, IO, etc.

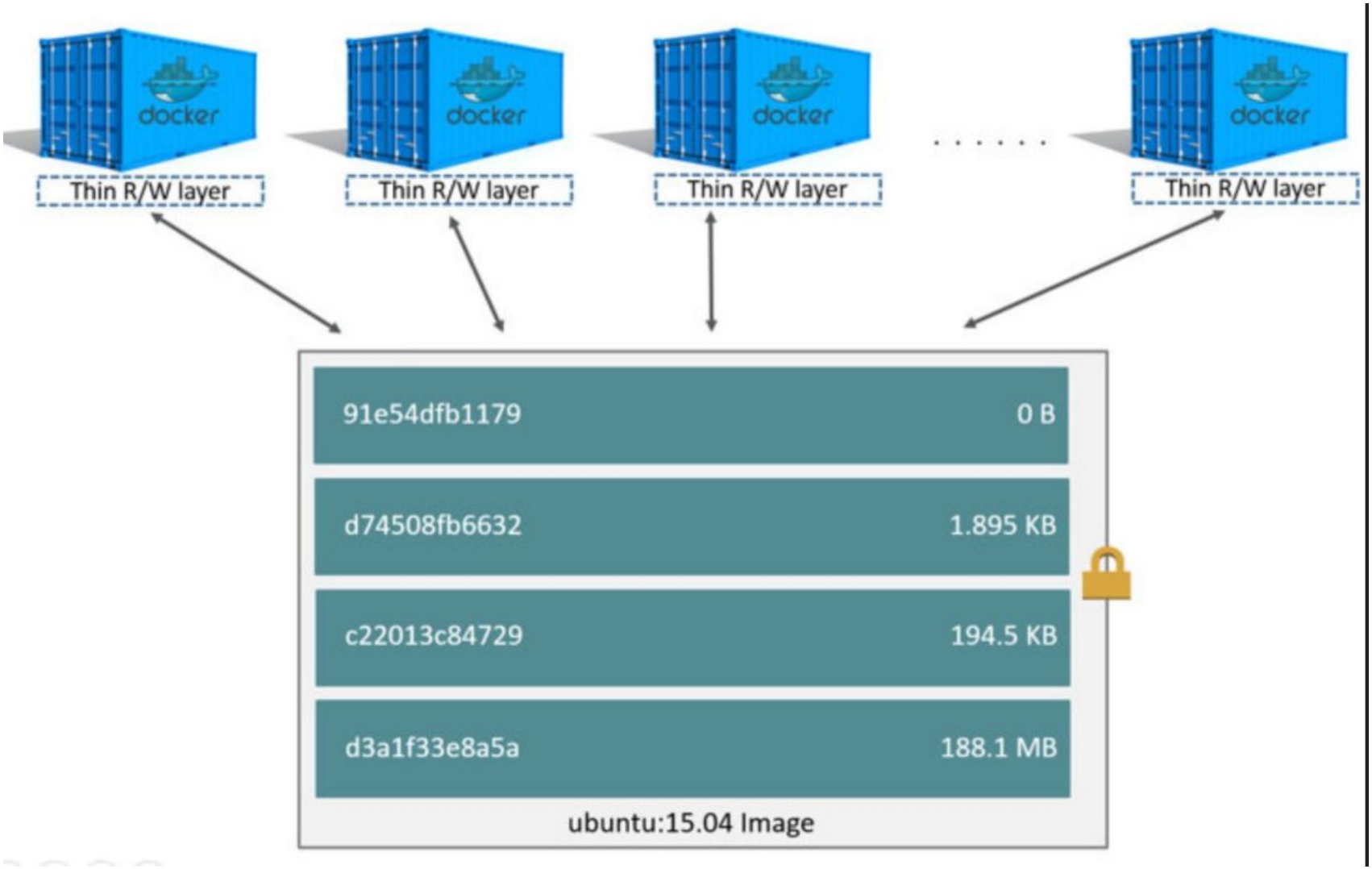
Контрольная группа (Linux) — Википедия

Механизмы контейнеризации: cgroups / Хабр

Docker unionfs



Docker unionfs



Docker union fs

Пример номер раз
Ссылка тут

Пример номер два
Ссылка тут

Dockerfile

Docker

- **ENV** - переменные окружения
- **ARG** - переменные во время сборки
- **COPY** - скопировать файл или папку
- **ADD** - скопироватƀ файл или папку, скачать по ссылке, разархивировать архив
- **EXPOSE** - документация

Docker

А как запустить-то?

- **CMD**
- **ENTRYPOINT**

Режимы работы:

shell

```
FROM alpine
ENTRYPOINT ping www.google.com
```

exec

```
FROM alpine
ENTRYPOINT ["ping", "www.google.com"]
```

Docker

Комбинированное использование

```
FROM alpine
ENTRYPOINT ["ls", "/usr"]
CMD ["/var"]
```

А если я хочу более сложную конструкцию для запуска?
Никто не запреûает использовать bash-скрипт

Docker

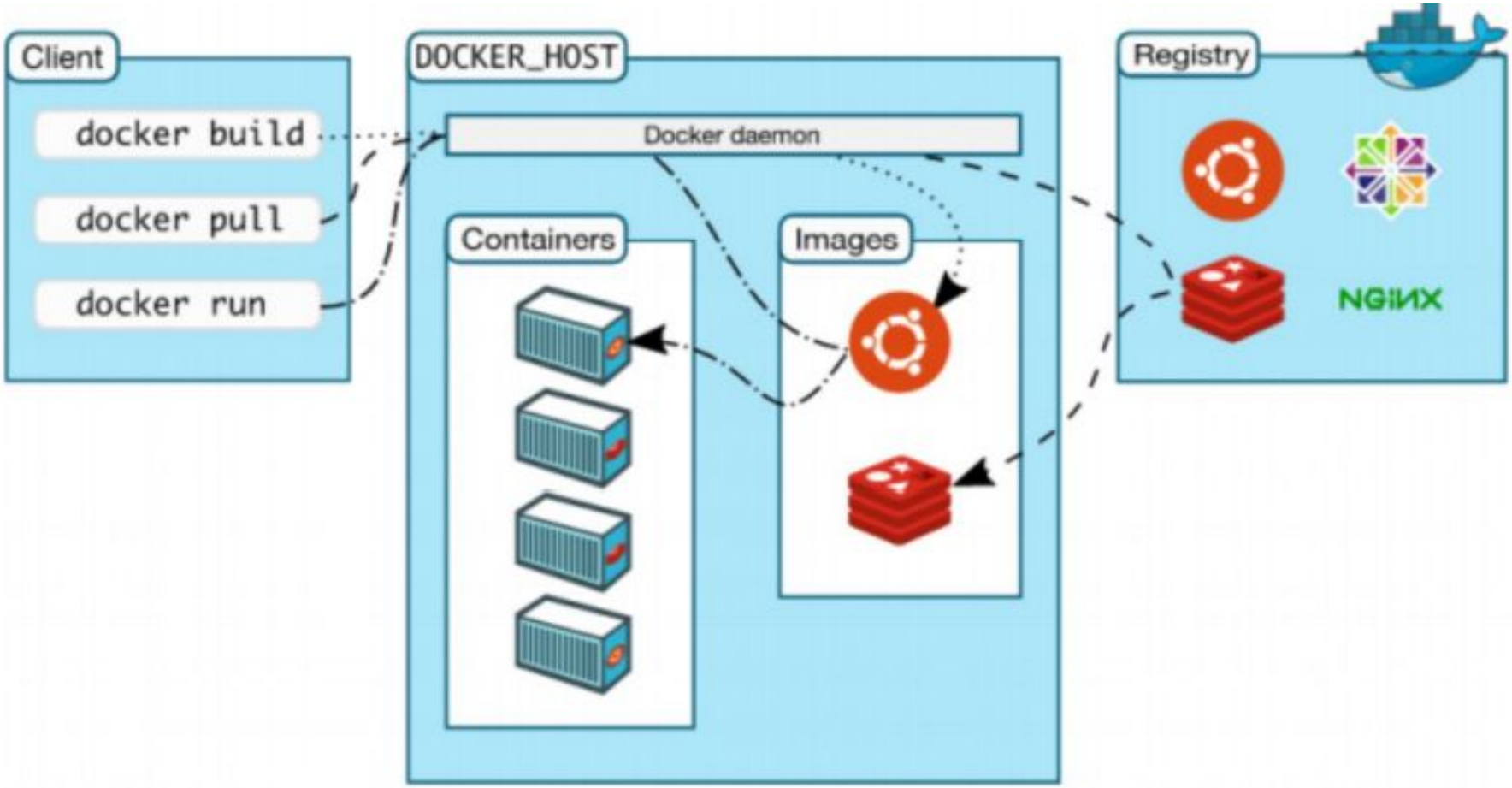
Daemon:

- предоставляет api
- управляет объектами
- взаимодействует с другими daemon`ми

cli:

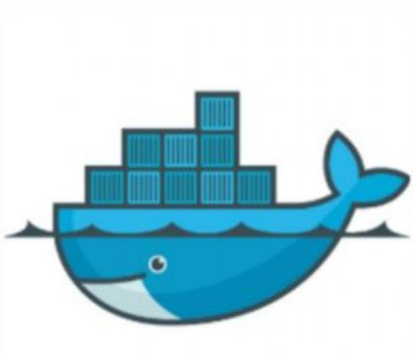
- принимает команды от пользователя
- взаимодействует с api docker daemon

Docker


Client
docker build
docker pull
docker run
DOCKER_HOST
Docker daemon
Containers
Images
Registry
NGINX

Docker

Так называемый приватный “репозиторий” для хранения image.
Docker hub, Private registry, Docker Store



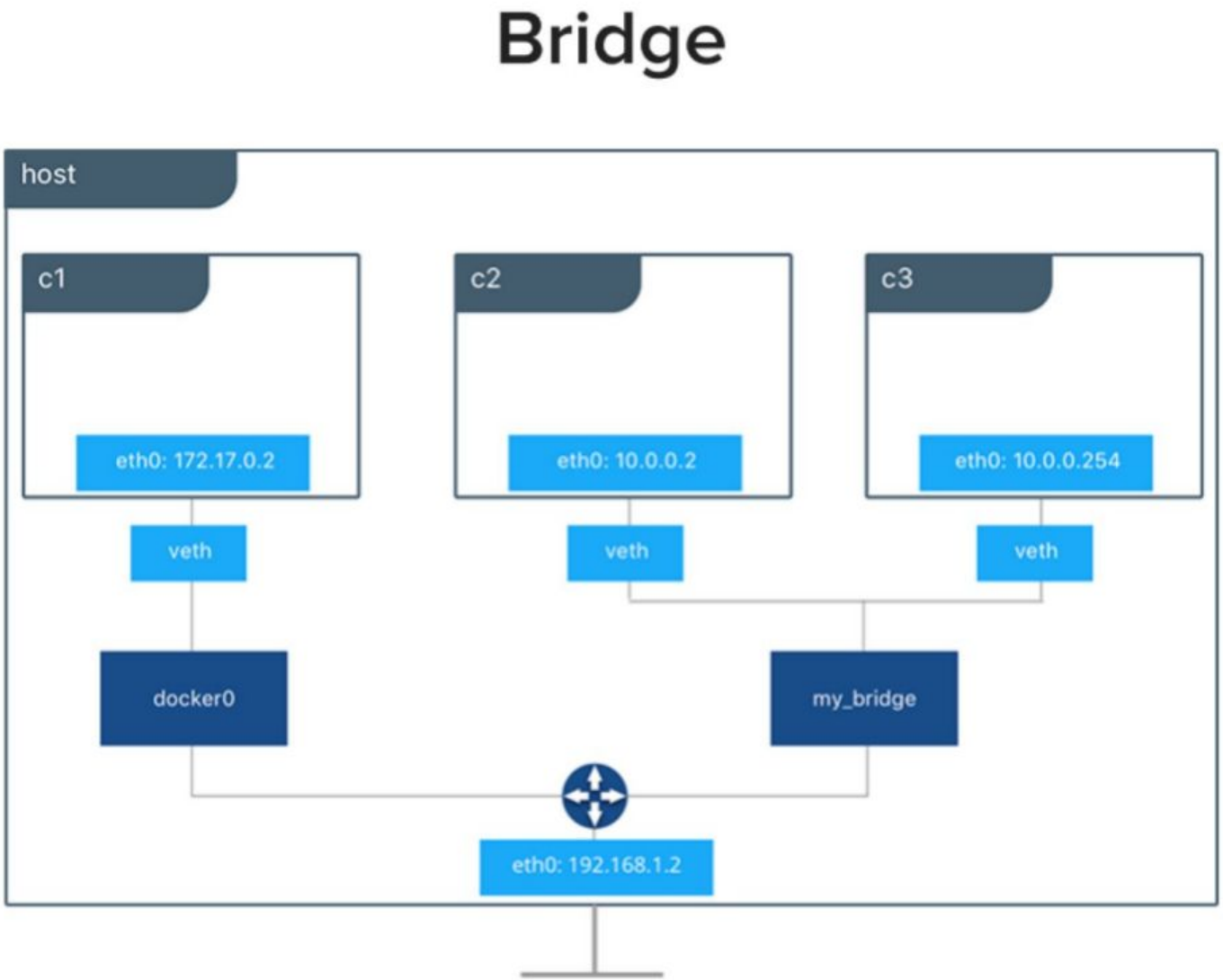
GitLab

Docker: сети

Docker сеть Bridge network

- Если нужно отделить контейнер или группу контейнеров
- Контейнер может быть подключен к нескольким Bridge сетям
- (без рестарта)
- Работает Service Discovery
- Произвольные диапазоны IP-адресов

Docker сеть Bridge network



Docker-compose

Docker-compose

Посмотрим на demo ^_^

Docker полезности

- docker ps - смотрим что есть
- docker df - сколько места съел докер
- docker stats - смотрим сколько ресурсов потребляет контейнер
- docker logs - логи контейнера
- docker inspect - информация по контейнеру
- docker … prune … - очистить (перед этим смотри доку)

Docker полезности

- 1. Смотрим офф документация
- 2. Не стесняемся пользоваться hadolint
- 3. Multistage наше все
- 4. Еще есть Dive
- 5. Поставим автокомплит для докера
- 6. Активно используем docker-compose
- 7. Смотрим что и как добавляем в образ
- 8. Не творим ДИЧЬ!!!!

Вопросы?

Ставим “+”,
если вопросы есть

Ставим “–”,
если вопросов нет

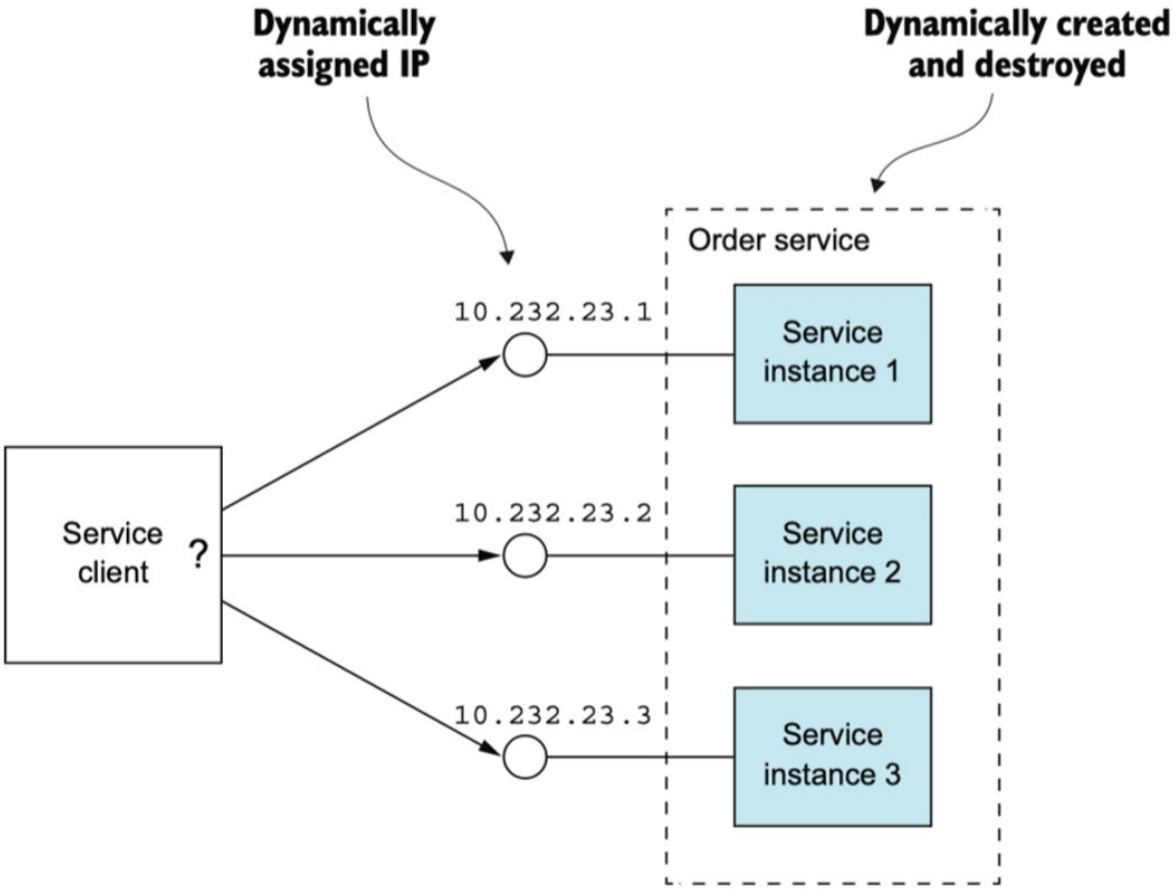
Consul

Service Discovery

1. **Как клиенту понять, где находится инстанс сервиса?**
2. **Почему нельзя хардкодить IP-адрес сервиса, к которому обращается клиент?**

Сроки выполнения: 3 мин

Service Discovery



Client-side Discovery

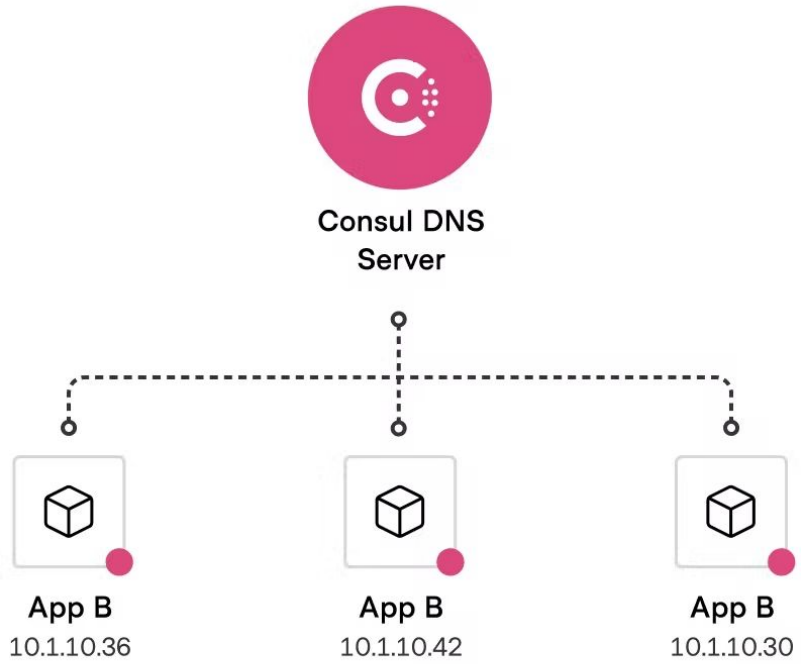
Примеры: Eureka, Consul, Zuul

- **Работает с несколькими системами оркестрации одновременно: k8s, standalone-сервисы, nomad и т.д.**
- **Зависит от поддержки языка программирования и фреймворка сервисов**

Пример: https://github.com/Netflix/eureka

```
@Autowired
private EurekaClient eurekaClient;

public void doRequest() {
    Application application
      = eurekaClient.getApplication("spring-cloud-eureka-client");
    InstanceInfo instanceInfo = application.getInstances().get(0);
    String hostname = instanceInfo.getHostName();
    int port = instanceInfo.getPort();
   //...
}
```

Consul



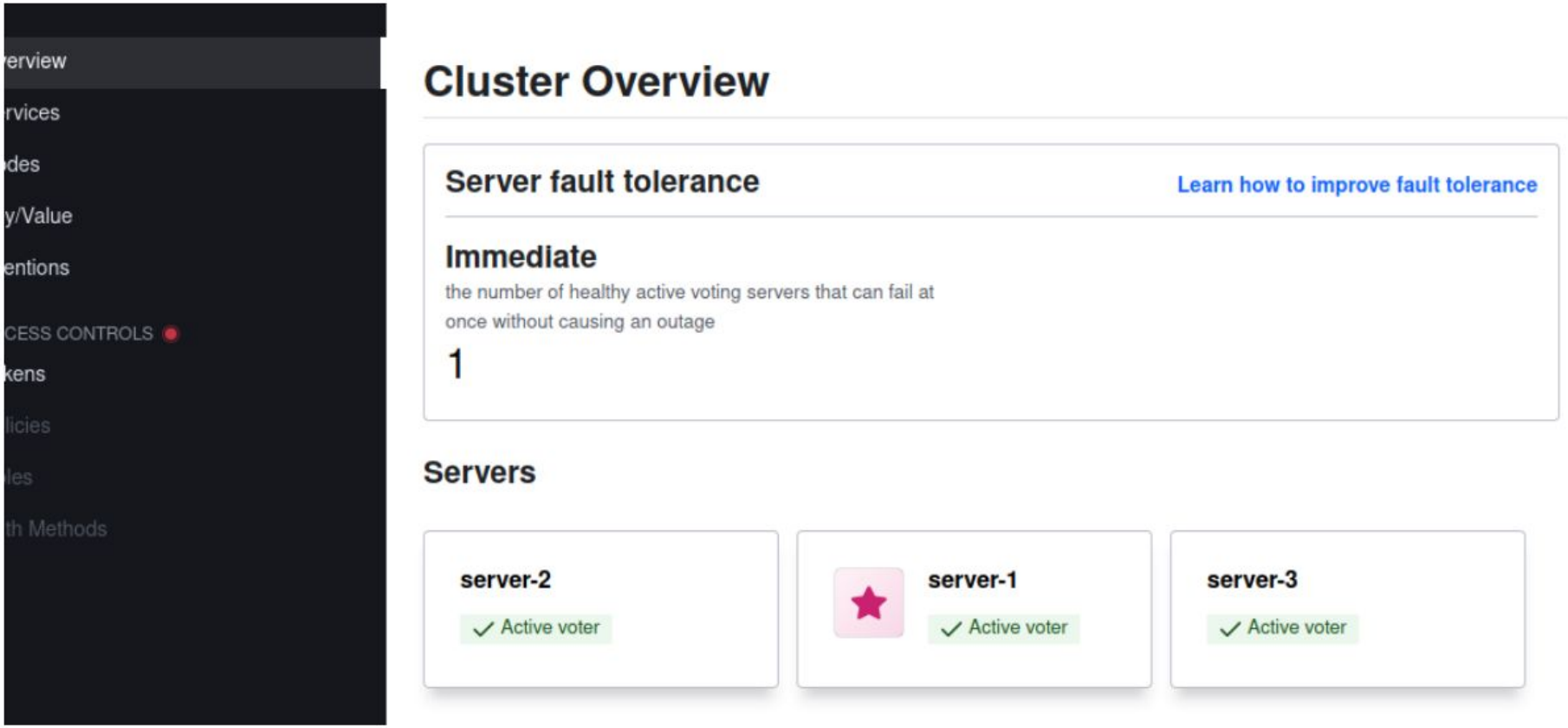
Consul example

Запущены 3 Consul сервиса при помощи докера

```
docker network create -d bridge consul

docker run --rm --net consul -p 8500:8500 -p 8600:8600/udp
--name=consul-server1 consul agent -server -node=server-1 -ui
-bootstrap-expect=3 -client=0.0.0.0

docker run --rm --net consul -p 8501:8500 -p 8601:8600/udp
--name=consul-server2 consul agent -server -node=server-2 -retry-join
consul-server1 -bootstrap-expect=3 -client=0.0.0.0

docker run --rm --net consul -p 8502:8500 -p 8602:8600/udp
--name=consul-server3 consul agent -server -node=server-3 -retry-join
consul-server1 -bootstrap-expect=3 -client=0.0.0.0
```

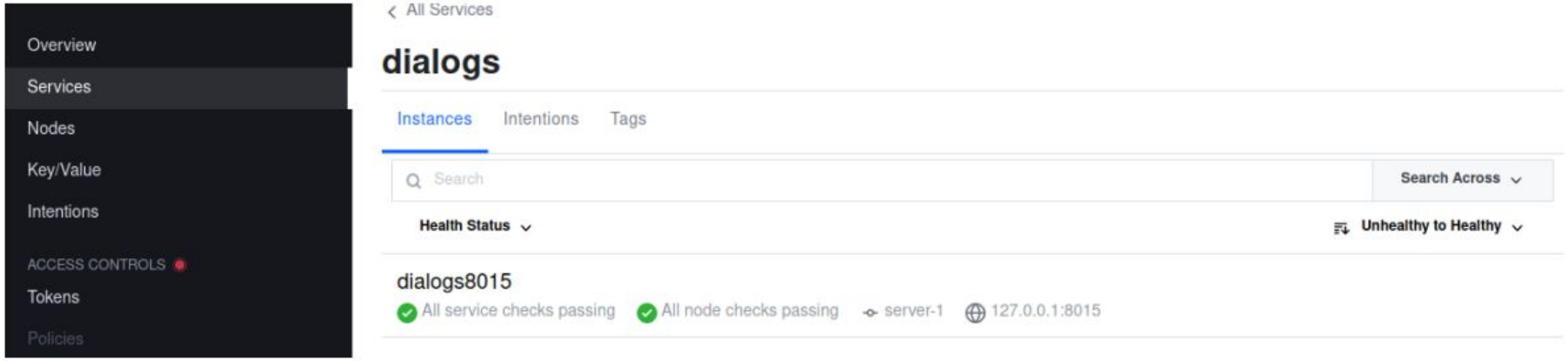
либо берем docker-compose отсюда: [Create a Secure Local Consul Datacenter with Docker Compose]

Consul example



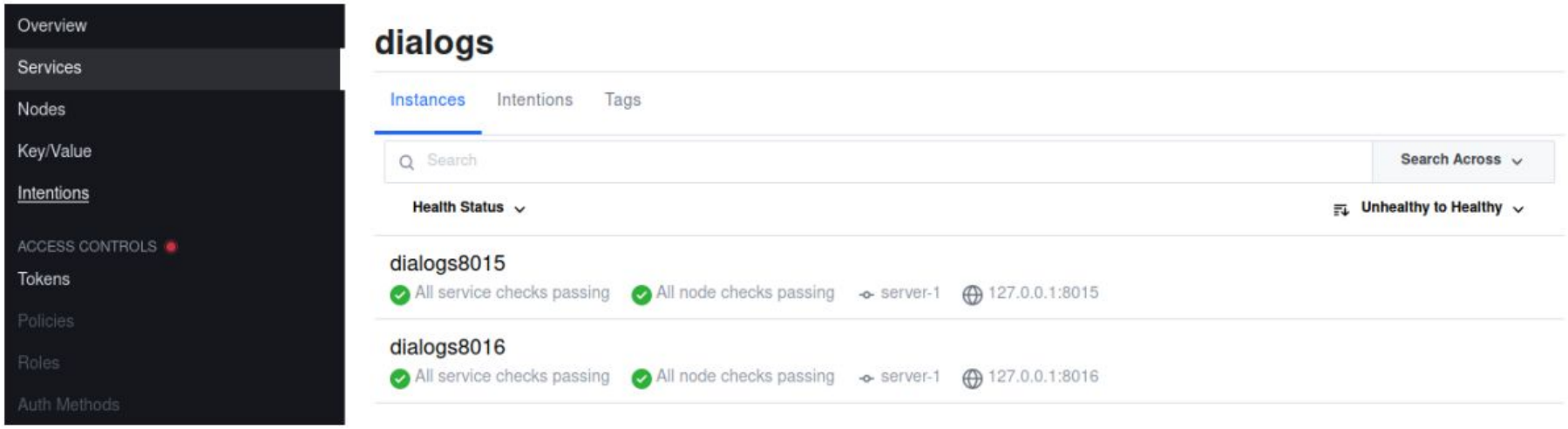
Consul example

Запускаем экземпляр сервиса диалогов



Consul example

Экземпляр доступен, запустим второй



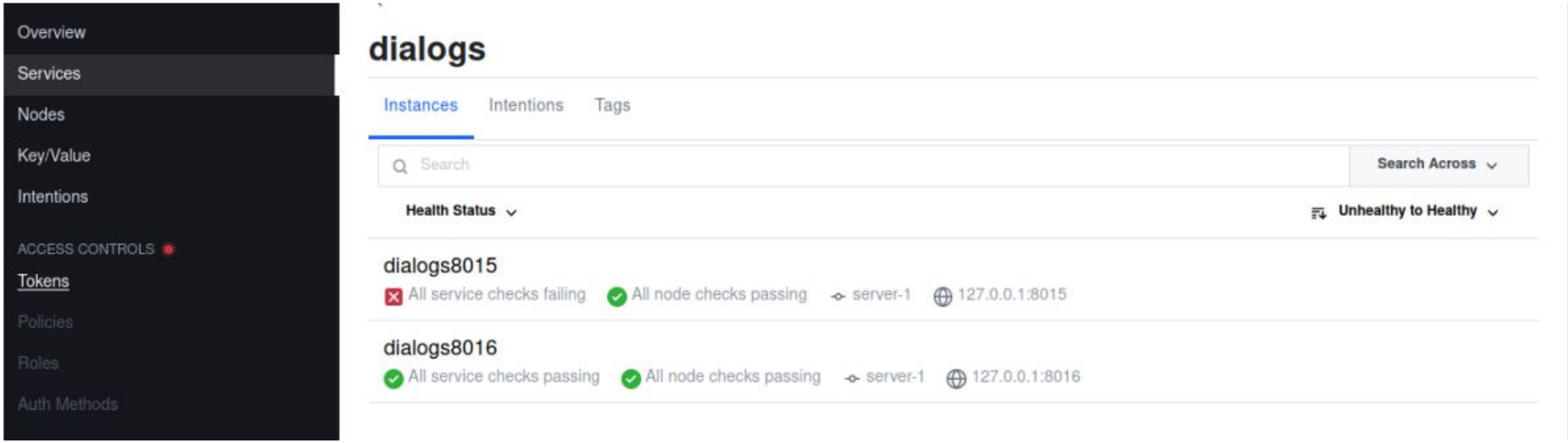
Consul example

Оба экземпляра доступны по запросу, который идет из резолвера:

```
 curl http://127.0.0.1:8500/v1/health/service/dialogs?passing=1
[{"Node":{"ID":"96cf8b07-964d-ae8f-b09b-8eb7142d8416","Node":"server-1","Address":"172.27.0.2","Datacenter":"dc1"
,"TaggedAddresses":{"lan":"172.27.0.2","lan_ipv4":"172.27.0.2","wan":"172.27.0.2","wan_ipv4":"172.27.0.2"},"Meta"
:{"consul-network-segment":""},"CreateIndex":8,"ModifyIndex":11},"Service":{"ID":"dialogs8015","Service":"dialogs
","Tags":[],"Address":"127.0.0.1","TaggedAddresses":{"lan_ipv4":{"Address":"127.0.0.1","Port":8015},"wan_ipv4":{"
Address":"127.0.0.1","Port":8015}},"Meta":null,"Port":8015,"Weights":{"Passing":1,"Warning":1},"EnableTagOverride
":false,"Proxy":{"Mode":"","MeshGateway":{},"Expose":{}},"Connect":{},"CreateIndex":55,"ModifyIndex":55},"Checks"
:[{"Node":"server-1","CheckID":"serfHealth","Name":"Serf Health
Status","Status":"passing","Notes":"","Output":"Agent alive and
reachable","ServiceID":"","ServiceName":"","ServiceTags":[],"Type":"","Interval":"","Timeout":"","ExposedPort":0,
"Definition":{},"CreateIndex":8,"ModifyIndex":8},{"Node":"server-1","CheckID":"service:dialogs8015","Name":"Servi
ce 'dialogs'
check","Status":"passing","Notes":"","Output":"","ServiceID":"dialogs8015","ServiceName":"dialogs","ServiceTags":
[],"Type":"ttl","Interval":"","Timeout":"","ExposedPort":0,"Definition":{},"CreateIndex":55,"ModifyIndex":56}]},{
"Node":{"ID":"96cf8b07-964d-ae8f-b09b-8eb7142d8416","Node":"server-1","Address":"172.27.0.2","Datacenter":"dc1","
TaggedAddresses":{"lan":"172.27.0.2","lan_ipv4":"172.27.0.2","wan":"172.27.0.2","wan_ipv4":"172.27.0.2"},"Meta":{
"consul-network-segment":""},"CreateIndex":8,"ModifyIndex":11},"Service":{"ID":"dialogs8016","Service":"dialogs",
"Tags":[],"Address":"127.0.0.1","TaggedAddresses":{"lan_ipv4":{"Address":"127.0.0.1","Port":8016},"wan_ipv4":{"Ad
dress":"127.0.0.1","Port":8016}},"Meta":null,"Port":8016,"Weights":{"Passing":1,"Warning":1},"EnableTagOverride":
false,"Proxy":{"Mode":"","MeshGateway":{},"Expose":{}},"Connect":{},"CreateIndex":86,"ModifyIndex":86},"Checks":[
{"Node":"server-1","CheckID":"serfHealth","Name":"Serf Health
Status","Status":"passing","Notes":"","Output":"Agent alive and
reachable","ServiceID":"","ServiceName":"","ServiceTags":[],"Type":"","Interval":"","Timeout":"","ExposedPort":0,
"Definition":{},"CreateIndex":8,"ModifyIndex":8},{"Node":"server-1","CheckID":"service:dialogs8016","Name":"Servi
ce 'dialogs'
check","Status":"passing","Notes":"","Output":"","ServiceID":"dialogs8016","ServiceName":"dialogs","ServiceTags":
[],"Type":"ttl","Interval":"","Timeout":"","ExposedPort":0,"Definition":{},"CreateIndex":86,"ModifyIndex":87}]}]
```

Consul example

Выключим первый сервис, Health-запросы перестанут приходить на Consul и он перестанет возвращать по запросу. Основное приложение сохранит работоспособность.



Вопросы?

Ставим “+”,
если вопросы есть

Ставим “–”,
если вопросов нет

Рефлексия

Рефлексия

С какими впечатлениями уходите с вебинара?

Как будете применять на практике то,
что узнали на вебинаре?

Полезные материалы

1. Книги Docker in Action, Docker in Practice, Docker up and running
2. How To Successfully Implement A Healthcheck In Docker Compose
3. Docker documentation: https://docs.docker.com/
4. Шпаргалка с командами Docker / Хабр

Заполните, пожалуйста, опрос о занятии по ссылке в чате

Спасибо за внимание!

Приходите на следующие вебинары

Золотов Дмитрий

Flutter & Kotlin Developer @ Yandex, DevOps, Fullstack Developer